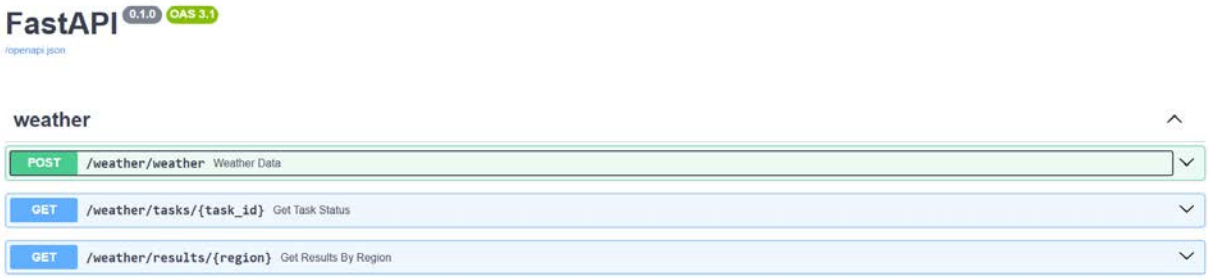
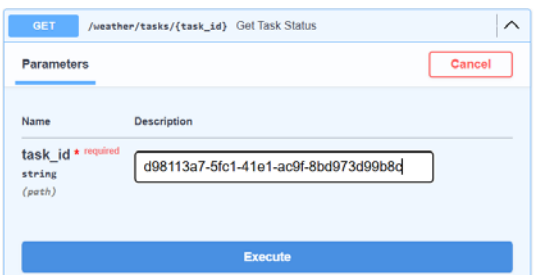
Вітаю!
Тут я опишу, як працює проект та по пунктах опишу виконані задачі.

Такий вигляд має API через Swagger



Вводимо в POST request назви міст, наприклад Південної Америки

Отримаємо



Отримаємо результати за цим ID

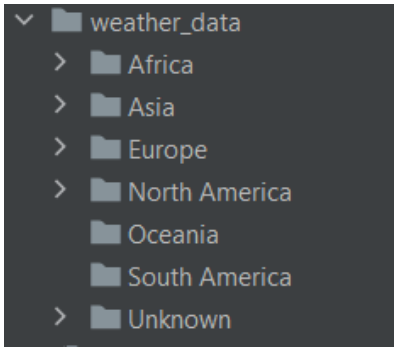
Response body


```
{
  "task_id": "d98113a7-5fc1-41e1-ac9f-8bd973d99b8c",
  "status": "completed",
  "result": {
    "results": {
      "Unknown": [
        {
          "city": "Rio de Janeiro",
          "temperature": 25.74,
          "description": "light rain"
        },
        {
          "city": "São Paulo",
          "temperature": 20.38,
          "description": "heavy intensity rain"
        }
      ],
      "South America": [
        {
          "city": "Buenos Aires",
          "temperature": 15.32,
          "description": "scattered clouds"
        }
      ]
    }
  }
}
```

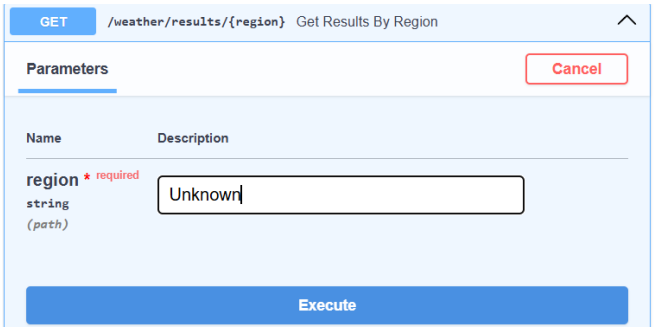

Download

Бачимо результат.

Buenos Aires був в нашій базі тому його відразу класифікувало по регіону. Міста які не визначені або неправильні відправляються в регіон Unknown.



Така структура розподілу міст по регіонам.



Response body


```
{
  "region": "Unknown",
  "data": [
    {
      "City": "R8",
      "Error": "City not found"
    },
    {
      "City": "Beijin",
      "Error": "City not found"
    },
    {
      "city": "Rio de Janeiro",
      "temperature": 25.74,
      "description": "light rain"
    },
    {
      "city": "São Paulo",
      "temperature": 20.38,
      "description": "heavy intensity rain"
    },
    {
      "city": "Kharkiv",
      "temperature": 1.36,
      "description": "broken clouds"
    },
    {
```


Download

Ось як бачимо тут неправильні або невизначені міста. Які ми можемо додавати для аналіз виправлень місь, або додавати їх в БД(в нашому випадку regions.json)


```
2024-12-26 11:05:25 weather-celery  | [2024-12-26 09:05:25,651: INFO/MainProcess] Task app.tasks.fetch_weather_task[d9811
3a7-5fc1-41e1-ac9f-8bd973d99b8c] received
2024-12-26 11:05:27 weather-celery  | [2024-12-26 09:05:27,070: INFO/MainProcess] HTTP Request: GET https://api.openweath
ermap.org/data/2.5/weather?q=Rio+de+Janeiro&appid=727a5ee3b7f40de78fcde72c9c1c8689&units=metric "HTTP/1.1 200 OK"
2024-12-26 11:05:27 weather-celery  | [2024-12-26 09:05:27,080: INFO/MainProcess] Successfully processed data for city: R
io de Janeiro
2024-12-26 11:05:27 weather-celery  | [2024-12-26 09:05:27,132: INFO/MainProcess] HTTP Request: GET https://api.openweath
ermap.org/data/2.5/weather?q=Sao+Paulo&appid=727a5ee3b7f40de78fcde72c9c1c8689&units=metric "HTTP/1.1 200 OK"
2024-12-26 11:05:27 weather-celery  | [2024-12-26 09:05:27,133: INFO/MainProcess] HTTP Request: GET https://api.openweath
ermap.org/data/2.5/weather?q=Buenos+Aires&appid=727a5ee3b7f40de78fcde72c9c1c8689&units=metric "HTTP/1.1 200 OK"
2024-12-26 11:05:27 weather-celery  | [2024-12-26 09:05:27,137: INFO/MainProcess] Successfully processed data for city: S
ao Paulo
2024-12-26 11:05:27 weather-celery  | [2024-12-26 09:05:27,155: INFO/MainProcess] Successfully processed data for city: B
uenos Aires
2024-12-26 11:05:27 weather-celery  | [2024-12-26 09:05:27,178: INFO/MainProcess] Task d98113a7-5fc1-41e1-ac9f-8bd973d99b
8c completed successfully.
2024-12-26 11:05:27 weather-celery  | [2024-12-26 09:05:27,184: INFO/MainProcess] Task app.tasks.fetch_weather_task[d9811
3a7-5fc1-41e1-ac9f-8bd973d99b8c] succeeded in 1.472985867000034s: {'Unknown': [{'city': 'Rio de Janeiro', 'temperature':
25.74, 'description': 'light rain'}, {'city': 'São Paulo', 'temperature': 20.38, 'description': 'heavy intensity rain'}],
 'South America': [{'city': 'Buenos Aires', 'temperature': 15.32, 'description': 'scattered clouds'}]}
2024-12-26 11:09:10 weather-api     | INFO:     192.168.32.1:41496 - "GET /weather/tasks/d98113a7-5fc1-41e1-ac9f-8bd973d9
9b8c HTTP/1.1" 200 OK
2024-12-26 11:15:22 weather-api     | Directory for region Unknown exists: True
2024-12-26 11:15:22 weather-api     | Files in directory: ['task_c5a61e68-6edb-4bae-bf6d-f53b6e849fe1.json', 'task_c7f4c5
33-f2a3-444b-9b85-f1b01ebe4a85.json', 'task_d98113a7-5fc1-41e1-ac9f-8bd973d99b8c.json', 'task_2e8bce7e-f93d-452e-b033-44c
81a10c06a.json', 'task_b0a05984-0ff1-49a4-bb62-238b641e9f49.json', 'task_15ecbef8-bc7d-480c-94f0-1662f0553efe.json']
2024-12-26 11:15:22 weather-api     | INFO:     192.168.32.1:39530 - "GET /weather/results/Unknown HTTP/1.1" 200 OK
```


Тут ми бачимо Log нашого API, де бачимо як обробляються данні на ті чи інші request`и.

```python
def validate_data(data):
    # Check if the required fields are present
    if not data.get("city") or not data.get("temperature") or not data.get("description"):
        return False

    # Example validation: temperature should be within the range of -50 to +50 °C
    temp = data.get("temperature")
    if temp is None or not (-50 <= temp <= 50):
        return False

    # If all conditions are met, the data is valid
    return True
```

Фільтрує міста з некоректними даними (наприклад, температура поза межами -50 до +50 °C).

Також доданий Readme.md file.

Requirements:

- Use Celery and Redis for asynchronous processing.
- API errors must be logged with error details.
- Code must be structured to scale for multiple regions.
- Additional implementation requirements:
  - Validate input using regular expressions.
  - Support API keys for multiple external services.

1.Було використано FastAPI – як основний frame work, Celery, Redis.

2. API записує в log помилки які виникають. Наприклад неправильно написані міста “prosto_nabir_simvoliw”

```
2024-12-26 11:32:46 weather-celery  | [2024-12-26 09:32:46,122: ERROR/MainProcess] City Prosto_Nabir_Simvoliw not found.
Error: Client error '404 Not Found' for url 'https://api.openweathermap.org/data/2.5/weather?q=Prosto_Nabir_Simvoliw&appi
d=727a5ee3b7f40de78fcde72c9c1c8689&units=metric'
2024-12-26 11:32:46 weather-celery  | For more information check: https://developer.mozilla.org/en-US/docs/Web/HTTP/Statu
s/404
```

3. Ми можемо далі додавати регіони в API покращувати функції.

4.1 Регулярні вирази в чистому вигляді не використовуються

```python
def correct_typos(city: str, typos_file: str = "app/typos/typos.json") -> str:
    # Correct common city name typos based on data from a JSON file
    # Load typos data from the file
    typos = load_typos(typos_file)

    # Normalize the city name (strip spaces, title case)
    city = city.strip().title()

    # Return the corrected city name if a match is found, otherwise return the original city
    return typos.get(city, city)
```

Функція готова до змін під любі вимоги, та додавання regular verbs.

4.2 Можемо підтримувати декілька Weather API просто змінюючи ключі доступу до API

```python
import os
from dotenv import load_dotenv

# Load environment variables from the .env file
load_dotenv()

# Configuration settings
API_URL_OPENWEATHERMAP = "https://api.openweathermap.org/data/2.5/weather"
API_KEY_OPENWEATHERMAP = os.getenv('API_KEY_OPENWEATHERMAP')
```

Також про проблеми які виникають та їх вирішення можна прочитати в Readme.md file так про те як запустити API через Docker.